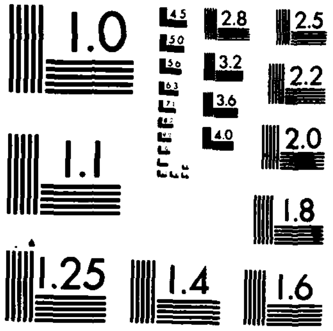
AD-A160 060 DEVELOPMENT OF AN ACCURATE ALGORITHM FOR EXP(BT)(U) 1/1
CALIFORNIA UNIV BERKELEY CENTER FOR PURE AND APPLIED
MATHEMATICS B N PARLETT ET AL. AUG 85 PAM-294
UNCLASSIFIED N00014-76-C-0013 F/G 9/2 NL

END
FILMED
DTIC



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU-OF STANDARDS-1963-A

2

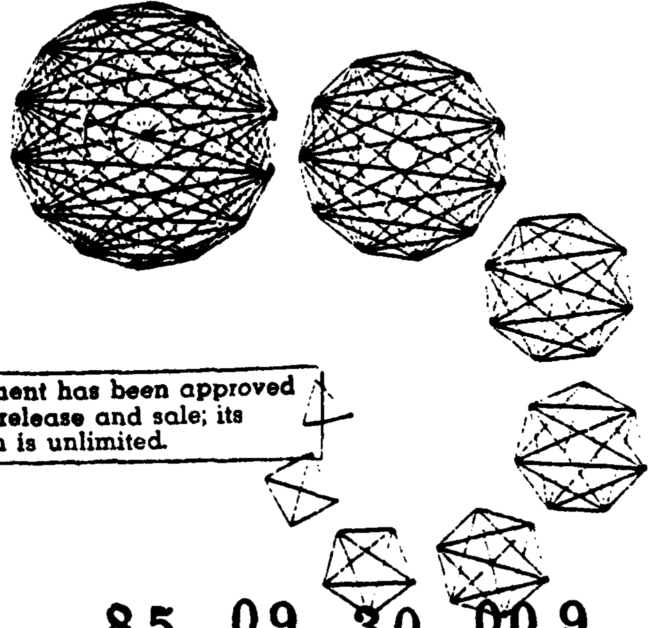
CENTER FOR PURE AND APPLIED MATHEMATICS
UNIVERSITY OF CALIFORNIA, BERKELEY

PAM-294

DEVELOPMENT OF AN ACCURATE ALGORITHM FOR EXP(Bt)

B.N. Parlett and K.C. Ng

August 1985



This report was done with support from the Center for Pure and Applied Mathematics. Any conclusions or opinions expressed in this report represent solely those of the author(s) and not necessarily those of the Center for Pure and Applied Mathematics or the Department of Mathematics.

DEVELOPMENT OF AN ACCURATE ALGORITHM FOR EXP(Bt)

B.N.Parlett[1] & K.C.Ng[2]

August 1985

CONTENTS

* This work was supported by the Office of Naval Research under contract N00014-76-C-0013.

[1] Math Dept.,UC Berkeley.

[2] Center for Pure & Applied Math, UC Berkeley.

1. Introduction

This document describes a program to compute the exponential of a given n by n matrix B multiplied by a scalar tau ~~t~~ that is to be thought of as representing time. The authors' ~~Our~~ primary goal has been to achieve as much accuracy as working precision permits without resorting to simulated higher precision. The final product is more complicated than they ~~we~~ anticipated at the outset. How these complications came to be accepted is the theme of this story. The cases we consider may be of interest to those who wish to use the matrix exponential in their work. We hasten to add that the code acts simply on simple cases.

This ~~Our~~ code is unlikely to be the last word on the computation of $\exp(Bt)$ since speed and simplicity deserve some recognition. For simplicity and generality this document ~~we~~ considers only complex matrices and only those whose order n is small enough that four n by n arrays fit comfortably in the fast store. In practice the user ~~we~~ expect to have $n<100$. Recall that unless B has special structure, such as being block diagonal, its exponential will have no zero elements. Additional keywords: applied mathematics; EISPACK computer program.

The rest of the paper is organized in the way indicated in the table of contents. We use standard conventions: capital letters for matrices, lower case for column vectors, and lower case Greek for scalars. In particular we switch to τ for time instead of t.

The designer of a program to invert a matrix has to decide what action should be taken when a given matrix is close to singular. Should an indication of sensitivity be given along with some output? Although the exponential of a matrix is always defined it may be exceedingly sensitive to tiny perturbations of the original matrix. Ideally a program such as ours should compute some measure of the *sensitivity* of the data. We have such a facility but, for the sake of brevity, we shall present it in another communication.

2. Background

Prior to the 1960's the task of computing the eigenvalues of a general matrix was considered difficult. Consequently a number of methods that avoid similarity transformations were tried. Examples are

(a) $\sum_{k=0}^{m} \frac{(B\tau)^k}{k!}$, for some suitable value of m,

(b) $r(B\tau)$, where r is a rational approximation to exp. The near-diagonal Pade approximations are favorites. See [Wragg and Davies] and the reference list of [Moler and van Loan].

(c) E^{2^j},where E is the result of (a) applied to $(B\tau)/2^j$.

A potential limitation of these methods is that they do not give the exact solution in exact arithmetic augmented with an exact exponential function. Consequently approximation error must be balanced against roundoff error. For example, straight forward use of the exponential series can yield some nasty surprises. See [Moler and van Loan].

The attraction of similarity transformations is that

$$f(GBG^{-1}) = Gf(B)G^{-1}$$

for any analytic function f. In particular, if B can be diagonalized in a stable way, say $GBG^{-1} = D = diagonal$, then

$$exp(Bt) = G^{-1}exp(Dt)G.$$

If cond(G) $(= \| G \| \cdot \| G^{-1} \|) < 100$ this is an attractive method. However not all matrices can be diagonalized. Since the effect of round off is proportional to cond(G) it is important to use only well conditioned similarities. That is one of the reasons why the Jordan canonical form is much less useful in matrix computations than in matrix theory. The similarities that put B in Jordan form may be arbitrarily ill-conditioned. The other count against this form when computing exponentials is worth mentioning. Sometimes it is overkill. Imagine a complicated Jordan structure associated with an eigenvalue -100 when there are other eigenvalues of magnitude -1. After exponentiation the contribution of the block associated with -100 to the whole will be negligible and there is no need to expend considerable effort to get that Jordan structure correct. To put it briefly, it is in general more difficult to compute the Jordan form than to compute the exponential.

Fortunately there is a canonical form that can always be obtained in a stable manner. Any square matrix may be put into upper triangular form by a sequence of unitary similarity transformations. Unitary matrices have the smallest possible condition number, namely 1, and are the analogues for complex matrices of the familiar orthogonal matrices such as plane rotations and Householder reflections. The reason that the Schur form S plays little role in matrix theory is that it is not unique. However once the order of the eigenvalues is fixed on the diagonal then the absolute value of each off diagonal element is fixed. That is good enough for practical work.

There are many other ways to approximate the exponential of a matrix. An excellent and informative description is given in [Moler and Van Loan]. To our knowledge none of them is suitable for the general case. We discuss an algorithm suggested by G.W.Stewart in a later section but it is, like ours, based on the Schur form.

3. An Ideal Algorithm and its Flaw

Nowadays, thanks to the QR algorithm (cf EISPACK routines), the reduction of a matrix to Schur (upper triangular) form is safe and easy. Only unitary similarity transformations need be used and this confers stability on the process. Moreover special 2 by 2 unitaries can be employed to swap adjacent diagonal elements of the Schur form S (these are necessarily eigenvalues of B) without destroying triangular form. Thus the eigenvalues may be put in any convenient order down the diagonal S. The attraction of this reduction is that once the diagonal elements of S have been exponentiated the upper triangular elements of $exp(S\tau)$ are rational functions of them and would be calculated exactly in exact arithmetic.

There are several ways to exponentiate a triangular matrix. Here is one. Impose a partitioning on S into block form that is as fine as possible subject to the constraint that *no two diagonal blocks have spectra that are close.* Thus all eigenvalues in a tight cluster must belong to the same block but a block may contain eigenvalues that are quite far apart. A picture is given below.

$$S = \begin{array}{cccccccc} X & X & X & \cdot & \cdot & \cdot & \cdot & \cdot \\ & X & X & \cdot & \cdot & \cdot & \cdot & \cdot \\ & & X & \cdot & \cdot & \cdot & \cdot & \cdot \\ & & & X & X & \cdot & \cdot & \cdot \\ & & & & X & \cdot & \cdot & \cdot \\ & & & & & X & X & X \\ & & & & & & X & X \\ & & & & & & & X \end{array}$$

It is easy to verify that $exp(S\tau)$ has the same block structure as S. Moreover there is a simple recurrence that permits the computation of the off diagonal blocks once the diagonal blocks of $E = \exp(S\tau)$ are known. This result was presented in [Parlett,1973]. For $j<k$,

$$S_{jj}E_{j,k} - E_{j,k}S_{kk} = E_{jj}S_{j,k} - S_{j,k}E_{kk} + \sum_{i=j+1}^{k-1} (S_{i,k}E_{j,i} - E_{i,k}S_{j,i}).$$

When $k=j+1$ then the recurrence yields $E_{j,j+1}$ as a function of the neighboring diagonal blocks of E. Similarly for each $k>j+1$ the blocks on the $(k-j)$th diagonal are determined by the blocks that are closer to the maim diagonal and lie in the corresponding row and column. Thus E can be built up from the diagonal outwards. Since the S_{jj} are triangular the elements of $E_{j,k}$ may be found sequentially from the bottom left element upwards. The partitioning ensures that these systems of equations are well conditioned. The procedure based on this recurrence is simple, stable, and fast. So the problem has been reduced to the computation of $\exp(\tau S_{jj})$, for each j, and deciding on the block structure.

Now the exponential series is an attractive method precisely when all the eigenvalues are clustered. For example, if all eigenvalues are equal then the series (expanded around the eigenvalue) must terminate at the nth term at the latest. In general we would expand about the mean μ of the eigenvalues in the block. Convergence is easy to monitor. One updates $(\tau S_{jj} - \mu I)^{k+1}$ from $(\tau S_{jj} - \mu I)^k$ and computes its norm in the process. If the norm is small enough one stops, otherwise one adds the next term.

This *ideal* algorithm will be accurate and efficient in many cases but it is not adequate for the general case. The flaw appears in the use of the exponential series for the diagonal blocks of $S\tau$. There is no guarantee that the eigenvalues of a block are close to their mean. In other words the conditions when the recurrence is accurate and the conditions when the exponential series converges satisfactorily do not cover all possibilities. Moreover the cases for which neither technique is satisfactory are quite numerous and not at all pathological. Our gradual realization of this unpleasant fact lead to a ten year retardation in our plans for producing a program for computing the matrix exponential. This also follows from the fact that point sets in the complex plane cannot always be *nested*. Moreover the larger the number of terms used in the series the greater the effect of roundoff and the greater the danger of having a hump. A hump occurs when one of the partial sums of the series is much larger than the final sum. The round off error is proportional to the norm of the greatest partial sum.

An ordered set of complex values $z_1,z_2,z_3,..$ is nested if, for any indices satisfying $i<j<k<l$, $|z_j - z_k| < |z_i - z_l|$. An example of a set that cannot be nested is a set of uniformly spaced points on a circle in the complex plane. In the algorithm given above if there are enough eigenvalues on the circle then they must all go in the same block but the radius of the circle may be large.

If the eigenvalues can be nested then such an ordering should be used for arranging them along the diagonal of S. However even if the eigenvalues are linearly ordered the spacing between them may dictate that they all should go int the same block despite a large gap between the first and last of them. Example II in the next section gives an instance in which the recurrence (using 1 x 1 blocks) is too inaccurate and the series is too slow. Note that for that example there is no better expansion point for this series than 0.

4. Illustrative Examples

The four examples in this section reveal different aspects of the task of computing the matrix exponential. These computations were done on a VAX/750 using single precision (complex) arithmetic. In the followings tables, cexphy[†] denotes the method we use in our matrix exponential program (see section 6 and 7).

Example I

$$Z = r\begin{bmatrix} 0 & 30 & 1 & 1 & 1 & 1 \\ -100 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & -6 & 1 & 1 \\ 0 & 0 & 500 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 200 \\ 0 & 0 & 0 & 0 & -15 & 0 \end{bmatrix}.$$

r	Method	Norm rel. error	Max. (Element-wise) rel. error	Timing	Remark
0.01	Taylor Series	2.11E-8	1.92E-7	.400 sec	11 terms
	cexphy	7.32E-7	7.86E-7	.250 sec	
0.1	Taylor Series	6.63E-7	1.22E-5	.867 sec	29 terms
	cexphy	8.77E-7	3.27E-6	.217 sec	
1	Taylor Series	2.72E15	2.51E16	4.08 sec	130 terms
	cexphy	5.60E-6	4.20E-5	.250 sec	
10	Taylor Series	-	-	-	hump overflow
	cexphy	3.15E-5	3.49E-4	.300 sec	
100	Taylor Series	-	-	-	hump overflow
	cexphy	3.35E-4	6.96E-3	.383 sec	

Table 4.1

The small integer matrix Z looks innocent but its spectrum consists of two triple eigenvalues near + 54.8i and -54.8i. Is this case difficult?

The answer is yes in real arithmetic but no in complex arithmetic! In the real case no block diagonalization is possible since the blocks have identical spectra. That leaves only the series on

[†] cexphy is the name of a subroutine in the Appendix.

the whole of Z for the "ideal" algorithm of Section 3. The diameter of Z's spectrum is 109.6 which suggests that many terms will be required in the series.

Now consider τZ. When τ is small (0.01) then the diameter is quite small too and only 11 terms are needed for the series to stagnate. As τ increases to 10 and beyond the computation is aborted by exponent overflow. A partial sum exceeds the narrow range of the VAX/780 (-38 to + 38 in decimal base) but even on other machines the computed sum, though very expensive, is worthless: it contains no correct digits. So overflow was merciful in this instance.

In complex arithmetic the two eigenvalues can be put into different blocks. These 3 by 3 blocks are easily exponentiated (Taylor series is almost the same as the Newton polynomial in this case) and the rest is filled in by the recurrence. Element relative accuracy is excellent until τ=1 but even for τ=100 at least half the digits in each element, even the smallest, are correct. The execution time grows only logarithmically with the norm of the matrix.

The reason that execution is quicker with τ=0.1 than with τ=0.01 is that in the latter case the eigenvalues are judged too close to separate and one 6 by 6 block is slower than two 3 by 3 blocks.

Example II

$$Z = \begin{bmatrix} -15i & -58 & 1 & 1 & \cdot & \cdot & 1 & 1 \\ & -14i & -54 & 1 & \cdot & \cdot & 1 & 1 \\ & & -13i & -50 & \cdot & \cdot & 1 & 1 \\ & & & \cdot & \cdot & \cdot & 1 & 1 \\ & & & & \cdot & \cdot & 1 & 1 \\ & & & & & 13i & 54 & 1 \\ & & & & & & 14i & 58 \\ & & & & & & & 15i \end{bmatrix}, \quad i=\sqrt{-1}.$$

Method	Norm rel. error	Max. (Element-wise) rel. error	Timing	Remark
Taylor Series	3.29E-3	7.42E-1	126. sec	58 terms
Recurrence	4.76E-2	2.68	1.15 sec	
cexphy	8.23E-6	5.85E-4	23.1 sec	

Table 4.2

This is a clear example in which neither the series nor the recurrence is adequate. Diagonalization also fails in the same way as the recurrence. The series is too slow (because the block's spectrum has diameter 30) and the recurrence is too inaccurate despite a separation of 1 between the eigenvalues. Our algorithm gets the smallest element of the exponential to 4 digits accuracy. It is 5 times quicker than the series but 20 times slower than the recurrence.

Example III

$$Z = \begin{bmatrix} -12 & 1 & & & & & & \\ & -11 & 1 & & & & & \\ & & -10 & 1 & & & & \\ & & & \ddots & \ddots & & & \\ & & & & 10 & 1 & & \\ & & & & & 11 & 1 \\ & & & & & & 12 \end{bmatrix}.$$

Method	Norm rel. error	Max. (Element-wise) rel. error	Timing	Remark
Recurrence	5.48E-8	1.20E-3	0.63 sec	
Taylor Series without shift	1.48E-7	1.48E+ 2	23.7 sec	52 terms
Taylor Series with shift=-12	1.42E-7	7.38E-7	32.3 sec	65 terms
cexphy	5.73E-8	2.34E-6	14.3 sec	

Table 4.3

This example illustrates two things. The first is the fairly well known rule that it is not always safe to expand the Taylor series around the mean of the eigenvalues of the matrix. However even with this shift Taylor series and all the other methods give perfect accuracy provided that you judge error relative to the norm of the exponential matrix. This permits small elements to be quite wrong. This may not matter in some applications where the exponential of the Schur form must be multiplied on each side by a fairly dense unitary matrix that will smear out the errors in the elements of $exp(\tau S)$. On the other hand our algorithm and the slow form of the series get all the elements correct. (In this example, the Taylor series run faster then expected because it took advantage on the bi-diagonal structure of Z.)

Example IV

Let $(\frac{a}{b})_r$ denote the quotient $\frac{a}{b}$ rounded to single precision floating point format.

$$Z = \begin{bmatrix} 0 & 2 & -2 & (8/3)_r & \cdot & \cdot & \cdot & (2^9/9)_r \\ & 0 & 2 & -2 & \cdot & \cdot & \cdot & \cdot \\ & & 0 & 2 & \cdot & \cdot & \cdot & \cdot \\ & & & \cdot & \cdot & \cdot & \cdot & \cdot \\ & & & & \cdot & \cdot & -2 & (8/3)_r \\ & & & & & 0 & 2 & -2 \\ & & & & & & 0 & 2 \\ & & & & & & & 0 \end{bmatrix} .$$

Method	Norm rel. error	Max. (Element-wise) rel. error	Timing	Remark
Taylor Series	1.01E-5	5.28E+ 1	1.05 sec	11 terms
cexphy	1.01E-5	5.28E+ 1	.553 sec	

Table 4.4

This example is a contrived one designed to defeat our program. The exponential of this matrix is very close to the Jordan block which has eigenvalue 1 on the diagonal and 2 on the superdiagonal. Both Taylor series and cexphy yield almost identical results but cexphy is faster. Although the error is fairly small relative to the norm of the solution our algorithm did not get the small elements with any accuracy. However our estimator of the sensitivity of this matrix to exponentiation, and indeed any reasonable estimator, declares that the small elements are exceedingly sensitive to any perturbations.

5. Relation to Block Diagonalization

In [7] Stewart proposed a way to compute the matrix exponential that is also based on the Schur form. He uses extra similarity transforms to reduce S to block diagonal form with as many blocks as is consistent with keeping the condition number of the transforms bounded. An implementation of Stewart's approach has been made by [8]. From the point of view of backward stability analysis there is little harm in allowing a condition number of 10 or even 100. He proposed to exponentiate the diagonal blocks that remain by using the series expansion around the mean of each block's eigenvalues. His algorithm differs from the *ideal* one described in the previous section only in the replacement of the recurrence formula by explicit reduction to block diagonal form.

We shall show below the surprising result that there are fundamental connections between the two approaches. Indeed, in one sense, they are equivalent. The attractive aspect of Stewart's approach is that the reduction is independent of τ. Consequently as τ changes it is only necessary to recompute the diagonal blocks. However the hidden difficulty is to decide on how large a condition number to allow for the additional similarities. Stewart wanted to keep the cutoff independent of the matrix or τ. The ideal algorithm uses dynamic grouping to determine the block structure. The criterion depends on the value of τ. The following example shows that it may not be wise to ignore τ.

Example.

$$\begin{pmatrix} 1 & 3 \\ 0 & 2 \end{pmatrix} = \begin{pmatrix} 1 & 3 \\ 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 \\ 0 & 2 \end{pmatrix} \cdot \begin{pmatrix} 1 & -3 \\ 0 & 1 \end{pmatrix}$$

Thus

$$\exp(\,\tau \begin{pmatrix} 1 & 3 \\ 0 & 2 \end{pmatrix}\,) = \begin{pmatrix} 1 & 3 \\ 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} e^{\tau} & 0 \\ 0 & e^{2\tau} \end{pmatrix} \cdot \begin{pmatrix} 1 & -3 \\ 0 & 1 \end{pmatrix}$$
$$= \begin{pmatrix} e^{\tau} & 3e^{2\tau}-3e^{\tau} \\ 0 & e^{2\tau} \end{pmatrix}$$

Diagonalization forces the (1,2) element to be computed by an explicit subtraction that will sacrifice information when τ is small. The recurrence will be just as bad. However by keeping the matrix as a 2 by 2 block one can use formulae or other techniques that yield the (1,2) element accurately for all τ. However it must be said that the error made using diagonalization or the recurrence, though avoidable and relatively large when τ is tiny, is nevertheless at the roundoff level compared with $|| exp(\tau B)||$. So, from the point of view of backward error analysis it is acceptable.

The next example shows that the requirement of keeping the condition number small can be too cautious. Both diagonalization and the recurrence are very accurate in exponentiating

$$\tau\cdot\begin{pmatrix} 10^{-4} & 1 \\ & 10^{-5} \end{pmatrix}, \text{ where } \tau = 10^{5}.$$

There is more to say on this topic. Block diagonalization is regarded as a simplification of the task that can be effected independent of τ However when τ is tiny then it is preferable to to treat the matrix S as one block rather than to implement extra similarity transformations. In these cases the Newton polynomial subroutine (see next section) will return

$$I+\tau S \quad \text{or} \quad I+\tau S+(1/2)\tau^2S^2$$

after computing the coefficients and checking that the remainder of them are negligible.

At the other extreme, when τ is large, then it is safe to use diagonalization even if the associated similarity transformations are ill conditioned. This is because the relative accuracy of the off diagonal terms of the exponential depends only on the eigenvalue separation, not on the other elements of S. See [3] for the analysis.

To see the relation between this reduction and the Recurrence it suffices to consider S partitioned into two blocks as shown

$$S = \begin{pmatrix} S_1 & S_{12} \\ 0 & S_2 \end{pmatrix}$$

where S_1 and S_2 are square. Provided that their spectra are disjoint a reduction can be effected by a simple similarity

$$\begin{bmatrix} I & R \\ 0 & I \end{bmatrix} \cdot S \cdot \begin{bmatrix} I & -R \\ 0 & I \end{bmatrix} = \begin{bmatrix} S_1 & 0 \\ 0 & S_2 \end{bmatrix}$$

where R satisfies

$$S_1R - RS_2 = S_{12}$$

The spectral norm of the reducing matrix is $\sqrt{1 + ||R||^2}$ and the same holds for its inverse. Consequently the condition number of the similarity is $1 + ||R||^2$. It would certainly be reasonable to use such transformations with R satisfying $||R|| < 3$ but $||R|| = 100$ would invite unnecessary loss of accuracy. Please note that our expressions are not mere bounds, they are exact. It should be pointed out at this point that there are better conditioned transformations that block diagonalize S but, unfortunately, they do not preserve the triangular form of S_1 and S_2. See [Demmel]

In practice the exponential $E = exp(S\tau)$ will be kept in factored form. So the (1,2) block of E will only be computed if E itself is wanted and in that case

$$E_{12} = E_1R - RE_2.$$

At this point it is convenient to introduce a little new notation. The right hand side of the above equation is a linear expression in R and this expression is sometimes called the Sylvester operator associated with E_1 and E_2. so we introduce

Definition.

$$Syl_E M = E_1M - ME_2.$$

Consequently

$$E_{12} = Syl_E R \quad \text{and} \quad Syl_S R = S_{12}.$$

On the other hand the recurrence (see Section 3) yields

$$Syl_S E_{12} = Syl_E S_{12}.$$

Now observe that E_i is a function of S_i, so E_i commutes with S_i for $i = 1,2$. Consequently Syl_E commutes with Syl_S. The verification is left as an exercise for the reader. Moreover, since the spectra of S_1 and S_2 are disjoint Syl_S is invertible and its inverse must also commute with Syl_E.

In other words

Via reduction: $E_{12} = Syl_E Syl_S^{-1} S_{12}$,

Via recurrence: $E_{12} = Syl_S^{-1} Syl_E S_{12}$.

The commutativity of the two *Syl* operators guarantees the same answer. In our case both *Syl* operators are triangular in nature and so E_{12} is obtained from S_{12} by two backsolves!

Of course the preceding analysis was in the context of exact arithmetic but it does give considerable insight into the properties of the actual algorithms executed in noisy arithmetic. For example, any error in E_i will get propagated to E_{12} through the operator Syl_S^{-1}. Hence the need for well separated spectra in S_1 and S_2.

Let us contrast block diagonalization with the recurrence. An advantage of reduction is that R is computed explicitly and so its norm will be known and can be judged during the computation. An advantage of the recurrence is that E_{12} is computed explicitly and uses inner products. So there is the possibility of using a function that employs accumulation of inner products with extra precision. However, given the partition, the two methods are essentially the same. In particular they both face the difficult task, to which the rest of this communication is addressed, of computing reliably E_i, $i=1,2$.

6. Outline of actual program

Justification, comments, and implementation tricks will be given in the sections that follow. The algorithm uses explicit similarities on a given complex matrix B. The goal is to form $exp(B\tau)$v for one or more vectors v and one or more values of the scalar parameter τ (time). It is convenient to keep $exp(B\tau)$ in factored form.

[1] Compute the Schur form S of B: $B=PSP^*$. $P^*=P^{-1}$, i.e. P is unitary. S is upper triangular. Eigenvalues are arranged down the diagonal in increasing order by real part.

[2] Impose a block structure on S such that the recurrence, given in Section 3, will determine the off diagonal blocks with high accuracy. Recall that the eigenvalues in the diagonal blocks may or may not be tightly clustered.

[3] Apply the following sequence of steps to each diagonal submatrix S_{jj} to form $E_{jj}=exp(\tau S_{jj})$. These blocks may be processed in turn or in parallel. In many instances only a subset of the following items will be used.

(a) Subdivide the eigenvalues of S_{jj} into tight clusters if possible. Here is an example. Let $i^2 = -1$.

$$S_{jj} = \begin{pmatrix} -0.1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ & 40.2i & \cdot & \cdot & \cdot & \cdot & \cdot \\ & & 40.4i & \cdot & \cdot & \cdot & \cdot \\ & & & -30.0i & \cdot & \cdot & \cdot \\ & & & & -30.3i & \cdot & \cdot \\ & & & & & 40.0i & \cdot \\ & & & & & & 0.1 \end{pmatrix}$$

Cluster 1: 40.4i, 40.2i, 40.0i

Cluster 2: -0.1, 0.1

Cluster 3: -30.0i, -30.3i

(b) Apply unitary similarities to S_{jj} to reorder the eigenvalues into the clusters found in (a). Say $S_{jj}=QR_{jj}Q^*$, where $Q^*=Q^{-1}$.

$$R_{jj} = \begin{bmatrix} 40.4i & * & * & * & * & * & * \\ & 40.2i & * & * & * & * & * \\ & & 40.0 & * & * & * & * \\ & & & -0.1 & * & * & * \\ & & & & 0.1 & * & * \\ & & & & & -30.0i & * \\ & & & & & & 30.3i \end{bmatrix} = \begin{bmatrix} R_{jj}^{(1)} & * & * \\ & R_{jj}^{(2)} & * \\ & & R_{jj}^{(3)} \end{bmatrix}$$

(c) Each $R_{jj}^{(k)}\tau$ is exponentiated using the Newton form of the interpolating polynomial on $R_{jj}^{(k)}\tau$'s eigenvalues. If there are fewer than 3 eigenvalues then explicit formulae are available (see the remark below).

$$exp(R_{jj}^{(k)}\tau) = e^{\lambda_1\tau} I + \sum_{m=2}^{p} \gamma_m \tau^{m-1} \prod_{\nu=1}^{m-1} (R_{jj}^{(k)} - \lambda_\nu I).$$

No extra storage is needed for the partial products. See later notes. The divided differences γ_m are computed by a mixture of a (scaling + squaring) technique, described as method (c) of Section 2, and the standard recurrence. Note that there may be no tight clusters and in such cases the polynomial is evaluated at τS_{jj}.

(d) Use the recurrence for the off diagonal blocks of R, if any.

(e) Reverse the similarities made in (b).

Remark. Formula for two by two triangular matrix. Let

$$B = \begin{bmatrix} b_1 & b_3 \\ 0 & b_2 \end{bmatrix}, \quad \omega = \tau(\frac{b_1 + b_2}{2}), \quad \psi = \tau(\frac{b_1 - b_2}{2}).$$

Then

$$\exp(\tau B) = \begin{bmatrix} e^{\tau b_1} & \tau b_3 \cdot e^{\omega} \cdot \sin(i\psi)/(i\psi) \\ 0 & e^{\tau b_2} \end{bmatrix}$$

[4] Apply the recurrence to compute the off diagonal blocks of E.

[5] For each given v compute $P(E(P^{*}\mathrm{v}))$.

7. Discussion of Program

[a] *Balancing.*

For our purposes there is a serious flaw in the EISPACK subroutine BALANC. It ignores the diagonal element when computing the norm of a row or column of a matrix. Although a diagonal similarity transformation using powers of the base of the arithmetic system causes no roundoff errors at all it may be ill-conditioned. In fact when balancing is most useful then the similarity will be ill-conditioned. The purpose of such transforms is to reduce the norm of the matrix and thereby to make the reduction to Schur form more accurate because norms are used, both explicitly and implicitly, to judge whether certain transforms are necessary. For the purpose of computing eigenvalues these transforms are beneficial but we are concerned about the effect of the back transformations, at the very end, when some function of the matrix has been computed. We have not made extensive tests with Balancing and consequently have not included it in our current implementation. Here is an example of an unfortunate transformation by BALANC. Given matrix B, let C be the balanced B.

$$B = \begin{pmatrix} 1 & 0 & 1E\text{-}10 & 0 \\ 0 & 2 & 0 & 1E\text{-}10 \\ 3 & 1 & 3 & 0 \\ 4 & 5 & 1 & 4 \end{pmatrix}.$$

After BALANC (only three digits are shown)

$$C = \begin{pmatrix} 1 & 0 & 8.19E\text{-}7 & 0 \\ 0 & 2 & 0 & 4.19E\text{-}4 \\ 3.66E\text{-}4 & 2.44E\text{-}4 & 3 & 0 \\ 4.77E\text{-}7 & 1.19E\text{-}6 & 9.77E\text{-}4 & 4 \end{pmatrix}$$

where

$$B = D{\cdot}C{\cdot}D^{-1}, \quad D = \begin{bmatrix} 1 & & & \\ & 2 & & \\ & & 2^{13} & \\ & & & 2^{23} \end{bmatrix}.$$

Apply Schur decomposition to B and C to obtain their Schur factors P_b, S_b and P_c, S_c. On a Vax 750 using single precision arithmetic, we have

$$|| B-P_b S_b P_b^* || = 7.78E{-}6,$$

$$|| B-DP_b S_c P_b^* D^{-1} || = 2.67.$$

Please note that if the diagonal elements were used in computing row and column norms in Balancing then B would be changed very little.

[b] *Criterion for Clustering.*

When the Schur form is real with real eigenvalues on the diagonal (and $\tau \geq 0$), [3] showed a way to determine the clustering that guarantees tight bound on the accumulated error in running the recurrence. We quote the result here.

Criterion R.

(i). Order the diagonals (eigenvalues) λ_i from left to right, i.e., $\lambda_i \leq \lambda_j$ if $i \leq j$.

(ii). If $\tau | \lambda_i - \lambda_j | \leq g(|i-j|)$ then λ_i and λ_j must go into the same cluster. Here

$$g(k) = k \cdot (1 + \frac{\ln k}{10}), \; k > 0.$$

It is not clear how to extend criterion R to complex numbers. A tentative solution, proposed in [3], is called *matrix argument reduction* We describe it briefly. It replaces τS by a different matrix G that has the same exponential as τS but all of whose eigenvalues lie within π of the real axis. Then Criterion R may be applied to G. This often works well but there are cases where G is not computed accurately enough in finite precision arithmetic. Further analysis shows that matrix argument reduction is equivalent to clustering the eigenvalues of S by their imaginary parts without regard to their real parts.

We have abandoned explicit argument reduction but have used it implicitly in designing our Criterion C for clustering. For each dimension k a bidiagonal matrix Z_τ ($z_{i,i} = \tau\cdot(i-k/2)$, $z_{i,i+1} = i$) is used to determine the smallest τ such that the recurrence formula still gives accurate answers, and define $g_k = \tau * k$. Then we interpolate these g_k by a second degree polynomial on positive integers k. Thus, we arrive at the following two level clustering criterion.

Criterion C.

(i). Order the eigenvalues according to their real parts (from negative to positive).

(ii). Apply Criterion R to determine clusters using

$$g(k)=0.2 + 2(k-1) + 0.03k^2.$$

(iii). For each cluster, reorder the eigenvalues by their imaginary parts. When these differ by less than π the eigenvalues are put into the same cluster.

We have concluded that the block structure in Step 3 must be a function of τ. The reason is independent of our use of the recurrence. The relative sensitivity of the (j,k) block of E depends on the separation of the eigenvalues of E_{jj} and E_{kk} and this will vary with τ. The plausibility of this decision may be seen from a consideration of extreme cases. When $\tau << ||B||$, say $10^{-8}||B||$, then it will be efficient to consider the matrix as one block and use a few terms of the interpolating polynomial. On the other hand when τ is large, but still small enough to make E computable, then the recurrence can be used for most of the off diagonal elements. Our code allows negative (even complex!) values for τ and a change of sign certainly warrants a new partitioning.

We had at least hoped to keep the ordering of the eigenvalues of the Schur form independent of τ but Step 3 of section 6 shows that we failed for the general complex case. However please note that this fact is hidden in Step 3. At the level of Steps 1,2,4, 5 the Schur form is fixed. By accepting the burden of a possible reordering at Step 3 we can obtain high relative accuracy whenever the data warrants it.

Let us consider an objection to Step 3. It seems likely that as τ varies the reordering in 3(a) of section 6 will turn out to remain the same. If this is the case then the unitary swaps could have been done initially at Step 2. The flaw in this argument is that the partitioning may be changing with τ and an ideal ordering may be very difficult to determine in advance.

[c] *The Interpolating Polynomial.*

We regard this procedure as an efficient form of the exponential series. It requires at most $m-1$ terms for an m by m matrix so a reasonable upper bound on the cost is known a priori: ($m^4/24$ + lower order terms) scalar multiplications. The price paid for this insurance is that the universal coefficients $1/k!$ are replaced by problem dependent divided differences γ_k. We have made a careful study of this problem in [3] and the conclusion is that, for exp, the whole set of coefficients can be evaluated with high relative accuracy in $O(m^2)$ scalar operations. The constant in O is between $\frac{1}{2}$ and $60 \cdot \log_2 m$. No extra storage is required.

[d] · *Computation of divided differences.*

It turns out that the γ_k form the top row of $\exp(Z)$, where Z is obtained from the given triangular matrix T by keeping the diagonal unchanged, putting 1's just above it, and putting to zero all the elements above the 1's. Thus Z is a bidiagonal matrix. In practice we prefer to compute scaled divided differences $\gamma_k(k-1)!$ and to scale down the partial products. See [3] for details.

8. Usage

The user is supposed to have an $n \times n$ complex matrix B, a complex τ, and an $n \times k$ complex matrix V. The task is to compute the $n \times k$ matrix U given by

$$U = \exp(\tau B)V.$$

By choosing $V=I$, the identity matrix, and $k=n$, the output will be $\exp(\tau B)$. The recomputation of U when τ and V are changed is easy and quick.

In order to invoke the subprograms the user must provide four 2-dimensional complex arrays:

$$B,P,E \text{ (each } n \times n\text{), and } V \ (n \times k).$$

In addition the program needs one real array w of dimension $10n$ and two integer arrays ip and idx of dimension n. At the end B will have been changed to S and V will have been overwritten with U.

In order to obtain U four subroutines must be called in the proper order.

```
      CALL  CSCHUR (B,P,low,igh,w,n,nb,np,ierr)
      CALL  CORDER (B,P,τ,w,n,nb,np)
      CALL  CEXPHY (B,E,w,ip,idx,τ,ovft,ierr,n,nb,ne)
      CALL  CFMULV (E,P,V,w,n,nb,ne,np,nv,k)
```

Here $low.igh,np,nb,ne,nv,n,k,ierr$ are integer variables. $ovft$ is the overflow threshold.

The recomputation of $\exp(\tau B)V$ with new values for τ and V is particularly efficient (approximately $5n^3$ for each τ). The calling sequence is as follows.

```
      CALL  CSCHUR (B,P,low,igh,w,n,nb,np,ierr)
      do 10 i=1, enough
         Get τ
         CALL  CORDER (B,P,τ,w,n,nb,np)
         CALL  CEXPHY (B,E,w,ip,idx,τ,ovft,ierr,n,nb,ne)
         Get V
         CALL  CFMULV (E,P,V,w,n,nb,ne,np,nv,k)
   10 CONTINUE
```

9. REFERENCES

[1] C.A.Bavely and G.W.Stewart, *An Algorithm for Computing Reducing Subspaces by Block Diagonalization,* SIAM Jour. Num. Anal. 16 (1979) pp.359-367.

[2] J. Demmel, *The Condition number of equivalence Transformations that Block Diagonalize Matrix Pencils,* SIAM Jour. Num. Anal. 20 (1983) pp.599-610.

[3] A. McCurdy, K.C. Ng and B.N. Parlett, *Accurate Computation of Divided Differences of The Exponential Function,* Mathematics of Computation, Volumn 43, Number 168, October 1984, pp501-528.

[4] C. Moler and C. van Loan, *Nineteen Dubious Ways to Compute The Exponential of a Matrix,* SIAM Review, Vol. 20, No. 4, October 1978, p801-836.

[5] B.N. Parlett, *A Recurrence Among the Elements of Functions of Triangular Matrices,* Linear Algebra Appl., 14(1976), pp.117-121.

[6] B.T. Smith, J.M. Boyle, B.S. Garbow, Y. Ikebe, V.C. Klema and C.B. Moler, *Matrix Eigensystem Routines - EISPACK Guide.* Lecture Notes in Computer Science, Vol 6, Springer, New York, 1974.

[7] G.W. Stewart, *Private communication.*

[8] A. Wragg and C.Davies, *Computation of the Exponential of a Matrix, Part I: Theoretical Considerations,* JIMA 11(1973)369-375, and *Part II: Practical Considerations,* JIMA 15(1975)273-278.

Security Classification

AD-A160 060

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author) University of California, Berkeley	2a. REPORT SECURITY CLASSIFICATION Unclassified 2b. GROUP

3. REPORT TITLE

Development of an accurate algorithm for exp(Bt)

4. DESCRIPTIVE NOTES (Type of report and inclusive dates)

Technical report, August 1985

5. AUTHOR(S) (First name, middle initial, last name)

B.N. Parlett and K.C. Ng

6. REPORT DATE August 1985	7a. TOTAL NO. OF PAGES 23	7b. NO. OF REFS 8

8a. CONTRACT OR GRANT NO. N00014-76-C-0013	9a. ORIGINATOR'S REPORT NUMBER(S)
b. PROJECT NO.	
c.	9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)
d.	

10. DISTRIBUTION STATEMENT

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY Office of Naval Research

13. ABSTRACT

This document describes a program to compute the exponential of a given n by n matrix B multiplied by a scalar τ that is to be thought of as representing time. Our primary goal has been to achieve as much accuracy as working precision permits without resorting to simulated higher precision. The final product is more complicated than we anticipated at the outset. How these complications came to be accepted is the theme of this story. The cases we consider may be of interest to those who wish to use the matrix exponential in their work. We hasten to add that the code acts simply on simple cases.

Our code is unlikely to be the last word on the computation of exp(Bt) since speed and simplicity deserve some recognition. For simplicity and generality we consider only complex matrices and only those whose order n is small enough that four n by n arrays fit comfortably in the fast store. In practice we expect to have $n < 100$. Recall that unless B has special structure, such as being block diagonal, its exponential will have no zero elements.

The rest of the paper is organized in the way indicated inthe table of contents. We use standard conventions: capital letters for matrices, lower case for column vectors, and lower case Greek for scalars. In particular we switch to τ for time instead of t.

(continued on back)

The designer of a program to invert a matrix has to decide what action should be taken when a given matrix is close to singular. Should an indication of sensitivity be given along with some output? Although the exponential of a matrix is always defined it may be exceedingly sensitive to tiny perturbations of the original matrix. Ideally a program such as ours should compute some measure of the sensitivity of the data. We have such a facility but, for the sake of brevity, we shall present it in another communication.

Legal Notice

This report was prepared as an account of work sponsored by the Center for Pure and Applied Mathematics. Neither the Center nor the Department of Mathematics, makes any warranty expressed or implied, or assumes any legal liability or responsibility for the accuracy, completeness or usefulness of any information or process disclosed.